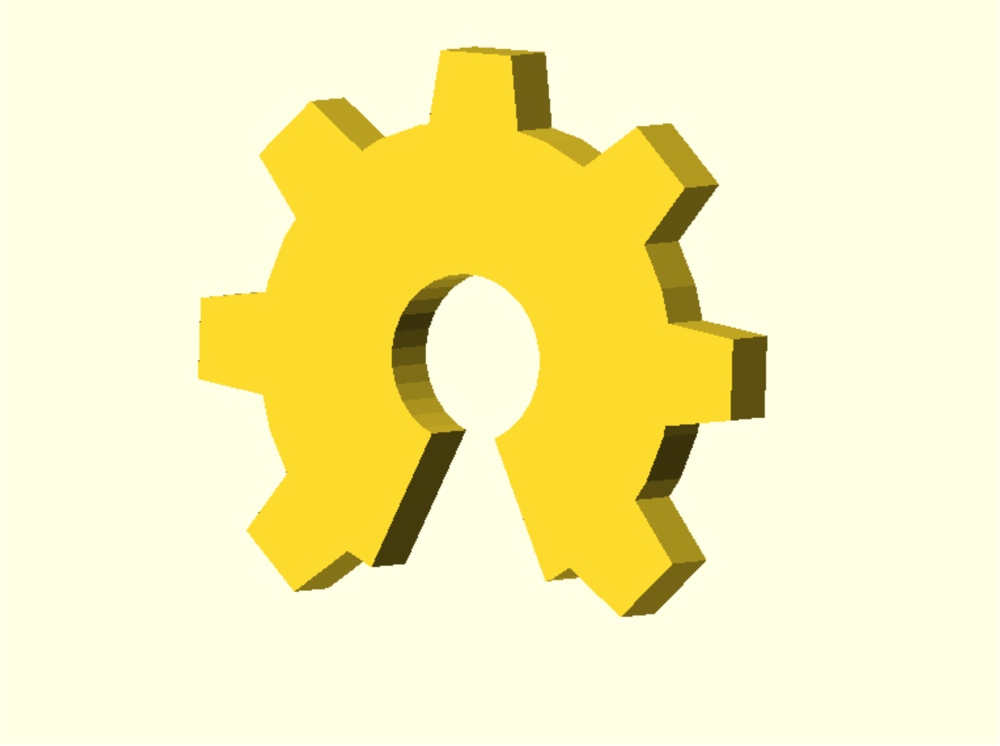
# Quick Introduction to OpenSCAD

Joshua M. Pearce
Department of Materials Science & Engineering and
Department of Electrical & Computer Engineering,
Michigan Technological University, Houghton, MI, USA

OpenSCAD
The Programmers Solid 3D CAD Modeller



MichiganTech
Michigan Technological University
Open Sustainability Technology
Research Group

# Make Everything Parametric

**Allows later scaling, changing and newbie customization**

**All numbers should be made variables**

Can use letters for simple designs **// but comment**

-advantages: simple equations

-disadvantage: big memory for large projects

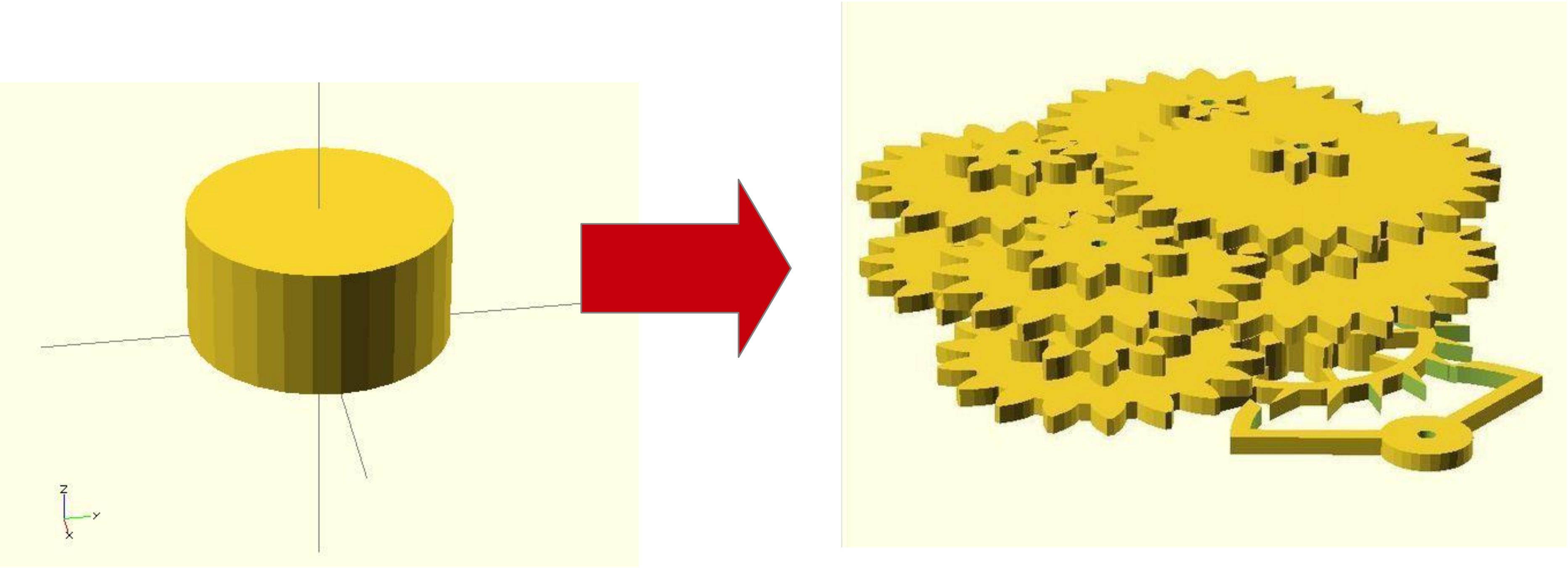
**Can use variable names describing it** // box_length

-advantages: no comments, can read the code in English

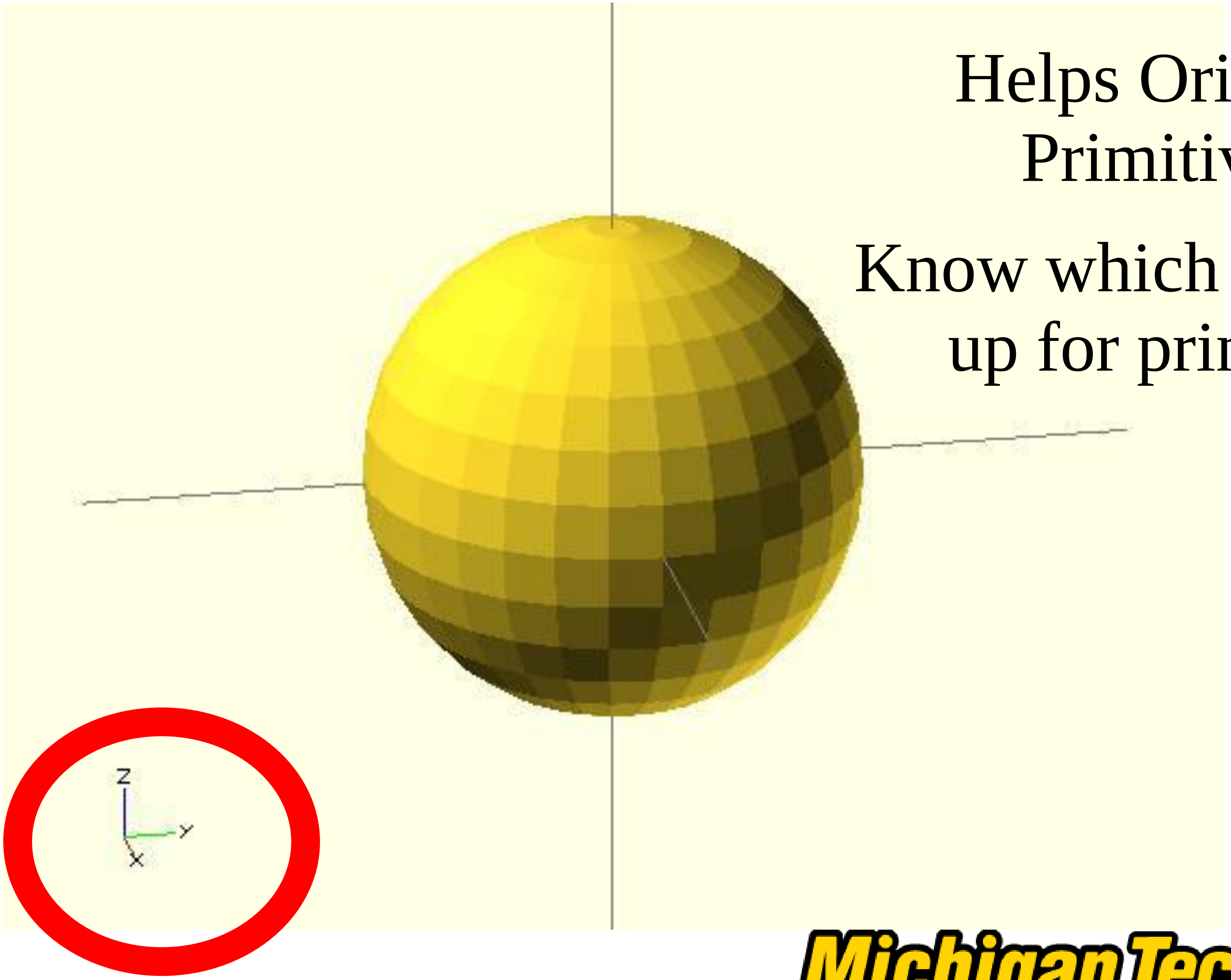
-disadvantage: big messy equations

# Design Using Primitive Shapes and Collecting Together

Simple → Complex



# When Designing: Show X-Y-Z



Helps Orient Primitives

Know which way is up for printing!

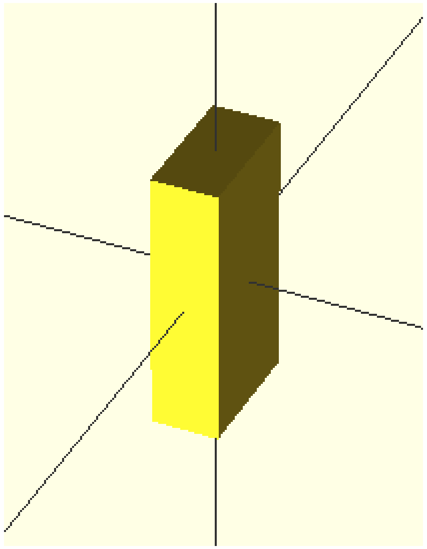
Michigan Tech

Michigan Technological University

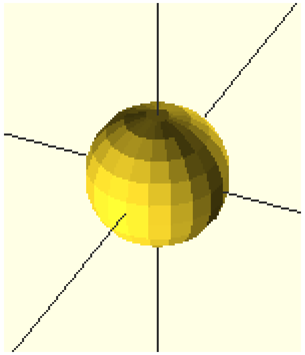
Open Sustainability Technology Research Group

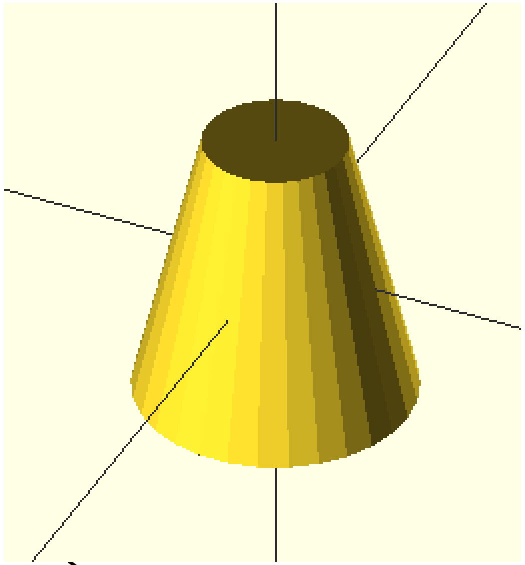
# Primitive Objects

```
a=5;
b=10;
c=20;
```



```
cube([a,b,c], center=true);
sphere(a, $fn=c);
//$fn is the resolution
cylinder(h = c, r1 = b, r2 = a, center = true);
```

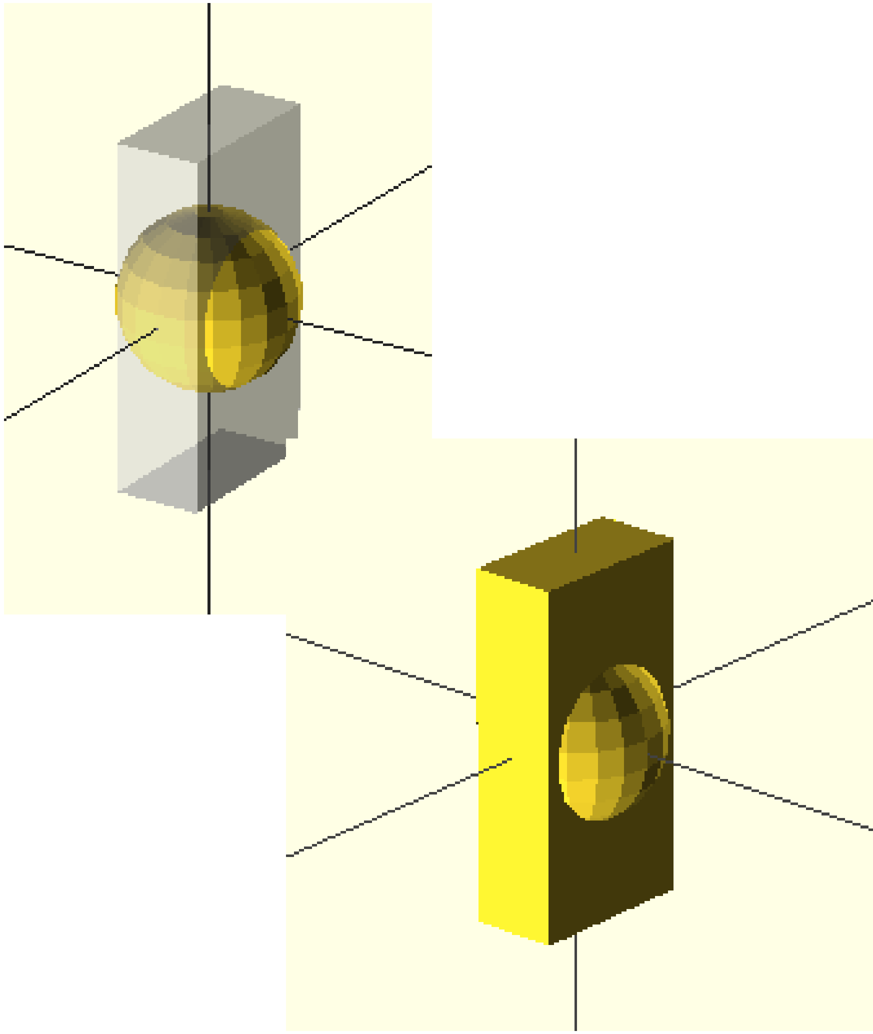




# Union Combining Primitives

“Try before you Buy”=%

```
union(){
%cube([a,b,c], center=true);
sphere(a, $fn=c);
}
```



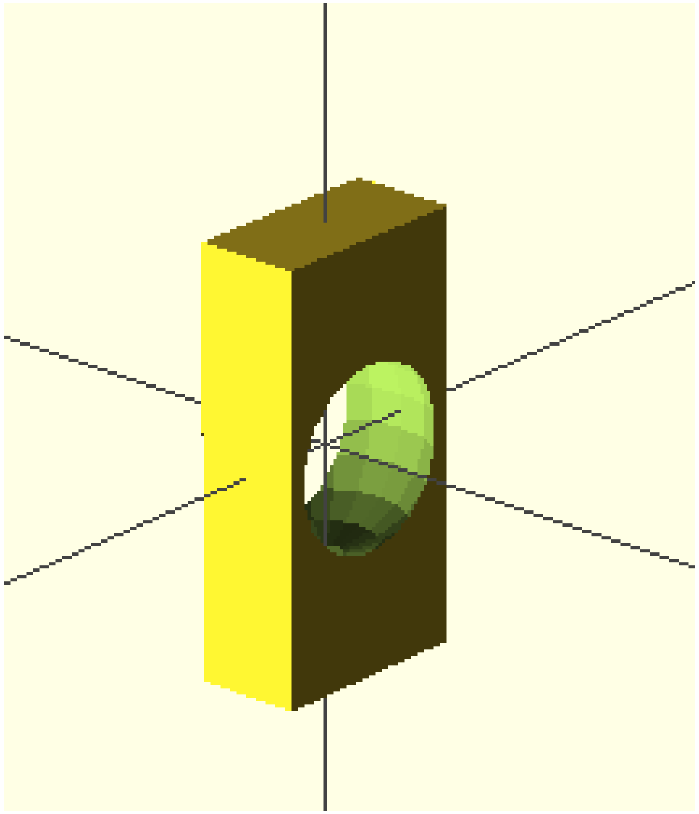
Michigan Tech

Michigan Technological University

Open Sustainability Technology Research Group

# Difference - Subtraction

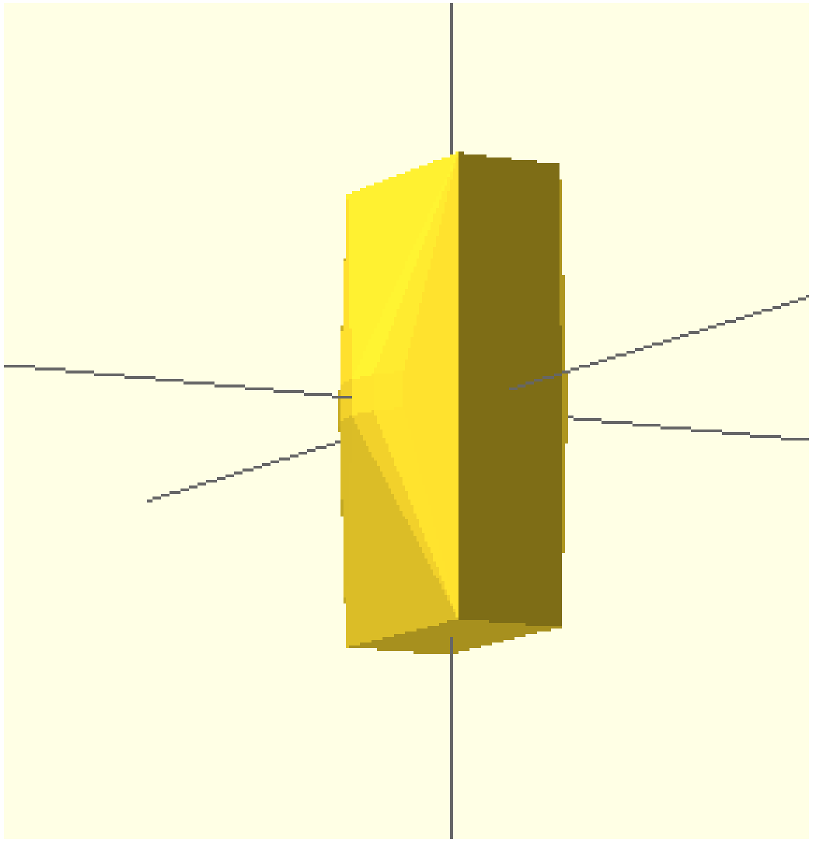
```
difference(){
cube([a,b,c], center=true);
sphere(a, $fn=c);
}
```



Michigan Tech
Michigan Technological University
Open Sustainability Technology Research Group

# Hull: Convex Hull of Child Nodes

```
hull(){
cube([a,b,c], center=true);
sphere(a, $fn=c);
}
```



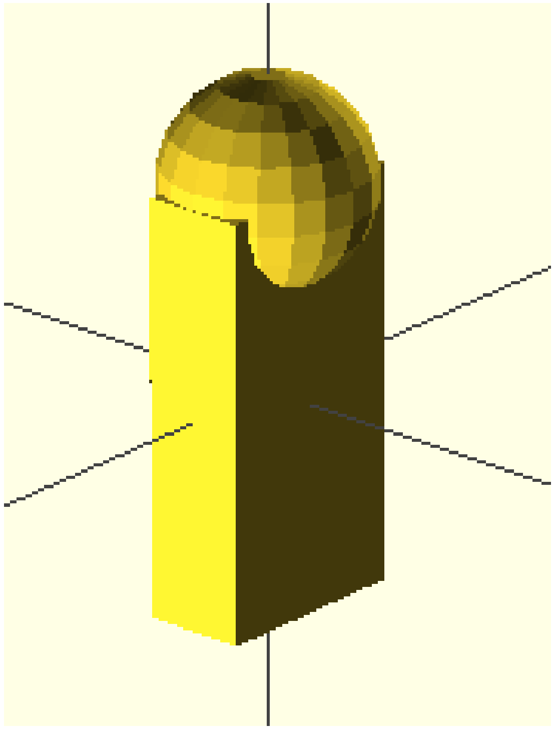
Michigan Tech

Michigan Technological University

Open Sustainability Technology
Research Group

# Translate: Moving Stuff Around

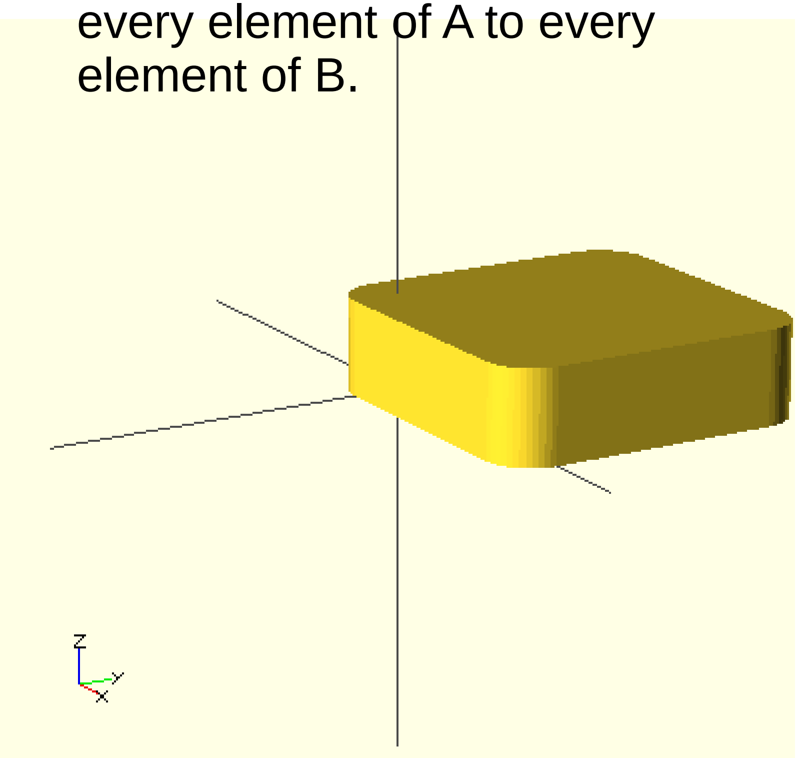
```
union(){
cube([a,b,c], center=true);
translate([0,0,b])sphere(a, $fn=c);
    }
```



# Rounded Corners: Minkowski

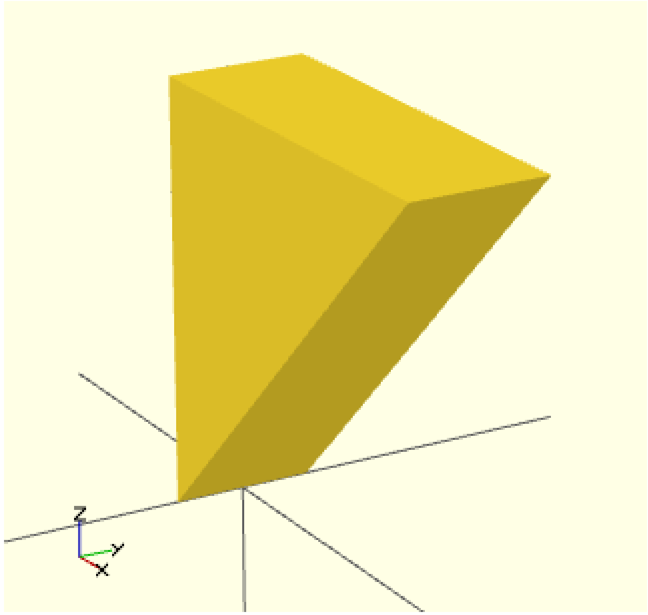
```
$fn=50;
minkowski() {
    cube([10,10,2]);
    // rounded corners
    cylinder(r=2,h=2);
}
```

Minkowski sums allow to add every element of A to every element of B.



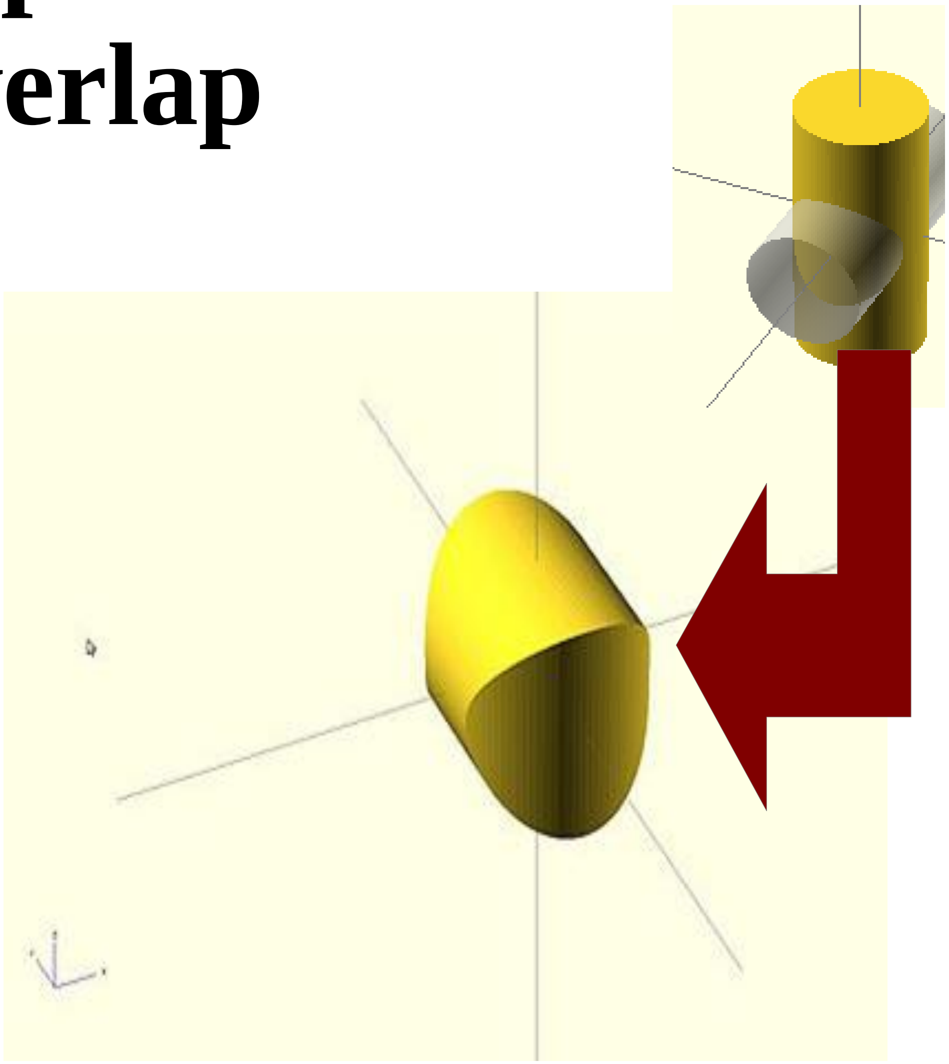
# ~~Hand Crafting: Polyhydron~~

```
polyhedron ( points = [[0, -10,
   60], [0, 10, 60], [0, 10, 0], [0,
   -10, 0], [60, -10, 60], [60, 10,
   60]],

triangles = [[0,3,2], [0,2,1],
   [3,0,4], [1,2,5], [0,5,4],
   [0,1,5],  [5,2,4], [4,2,3], ]);
```



# Intersection : Keeps All Portions That Overlap

```
intersection() {
cylinder (h = 4, r=1, center = true, $fn=100);
rotate ([90,0,0]) cylinder (h = 4, r=0.9, center = true, $fn=100);
}
```



# Make Each Completed Component a Module

**Allows for more complex design**

**Clears the work space as modules are not shown unless called**

**Syntax:**

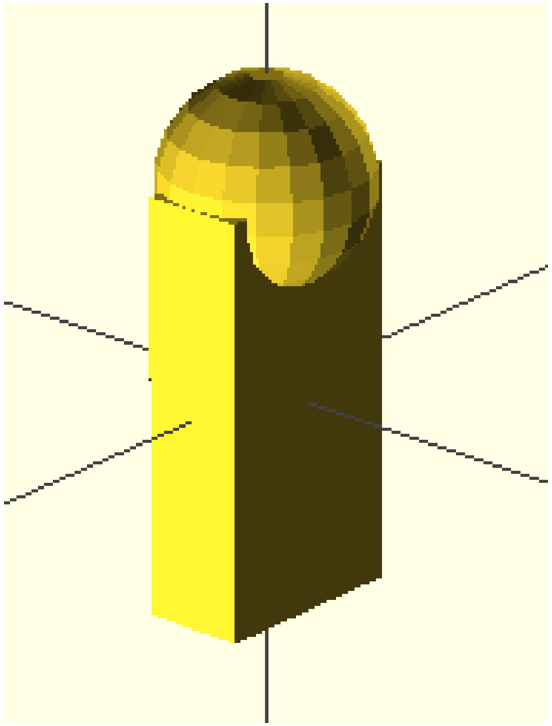
```
module example(){ put your module scad here }
```

**Call it by:**

```
example();
```

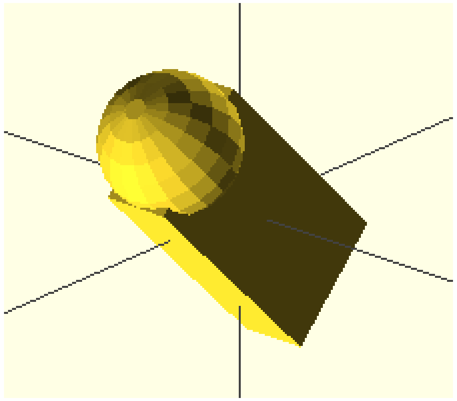
# Modules

```
module example(){

union(){

cube([a,b,c], center=true);

translate([0,0,b])sphere(a,
   $fn=c);

}

}


example();
```

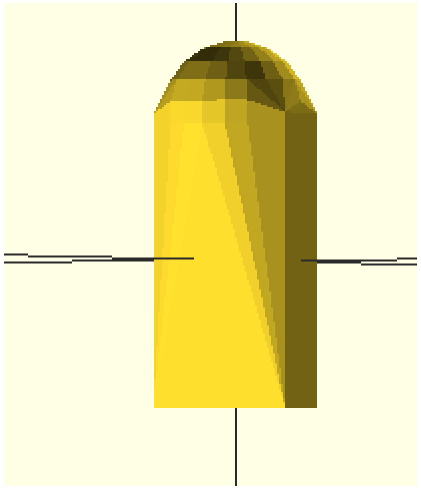


# Manipulate Your Module

```
rotate([45,0,0])example();
```



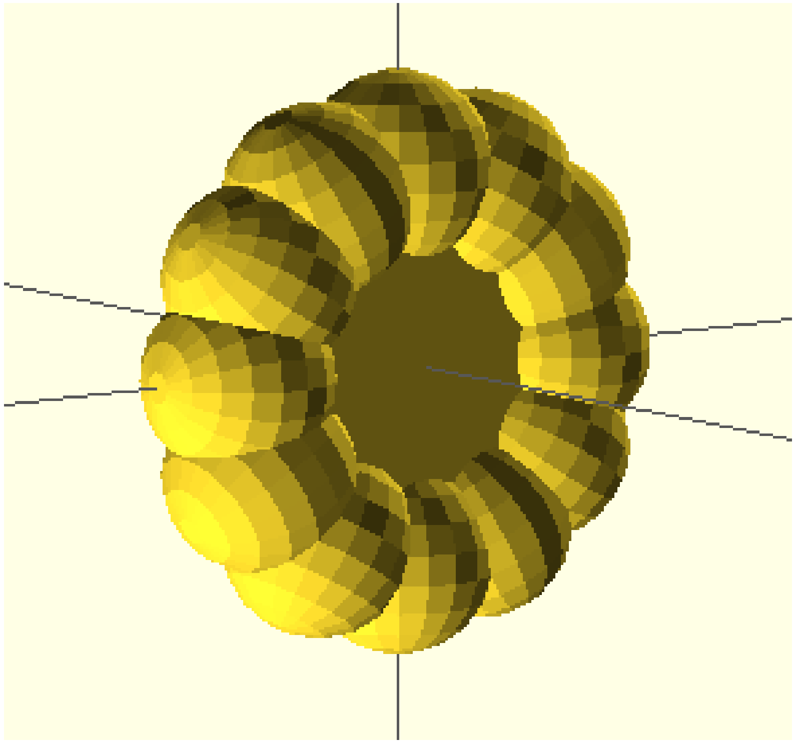
```
hull() {
example();
}
```



Add, subtract modules etc.

# For Repetitive Tasks Use Loops

```
for (i = [1:12])
{
    assign (angle = i*30)
    {
        rotate(angle, [1,0,0])
  example();
    }
}
```

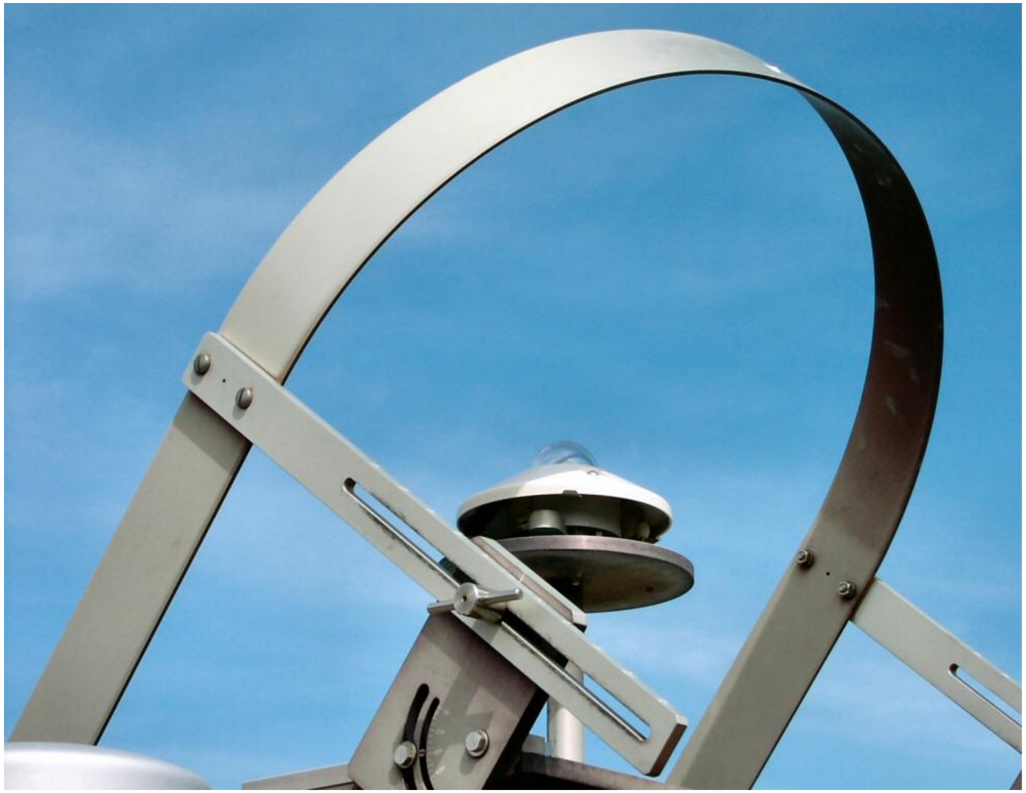


Michigan Tech

Michigan Technological University
Open Sustainability Technology
Research Group

# Applying OpenSCAD to Science

Shadow Band Pyranometer

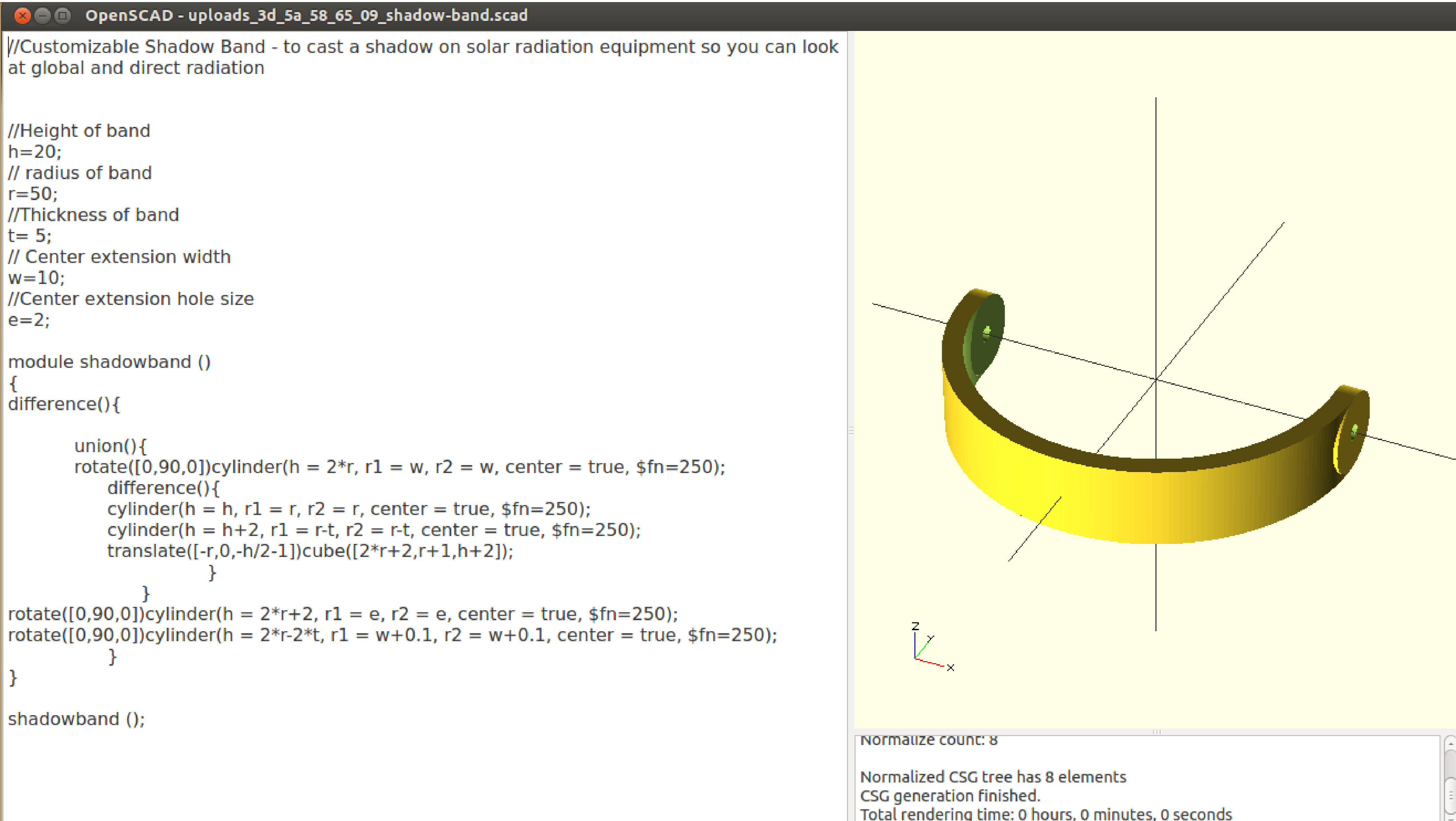


Michigan Tech

Michigan Technological University

Open Sustainability Technology Research Group

# Customization is Easy : OpenSCAD Parametric Shadowband for Pyranometer



```
//Customizable Shadow Band - to cast a shadow on solar radiation equipment so you can look at global and direct radiation


//Height of band
h=20;
// radius of band
r=50;
//Thickness of band
t= 5;
// Center extension width
w=10;
//Center extension hole size
e=2;

module shadowband ()
{
difference(){

        union(){
        rotate([0,90,0])cylinder(h = 2*r, r1 = w, r2 = w, center = true, $fn=250);
            difference(){
            cylinder(h = h, r1 = r, r2 = r, center = true, $fn=250);
            cylinder(h = h+2, r1 = r-t, r2 = r-t, center = true, $fn=250);
            translate([-r,0,-h/2-1])cube([2*r+2,r+1,h+2]);
                        }
                }
rotate([0,90,0])cylinder(h = 2*r+2, r1 = e, r2 = e, center = true, $fn=250);
rotate([0,90,0])cylinder(h = 2*r-2*t, r1 = w+0.1, r2 = w+0.1, center = true, $fn=250);
            }
}

shadowband ();
```

```
Normalize count: 8

Normalized CSG tree has 8 elements
CSG generation finished.
Total rendering time: 0 hours, 0 minutes, 0 seconds
```

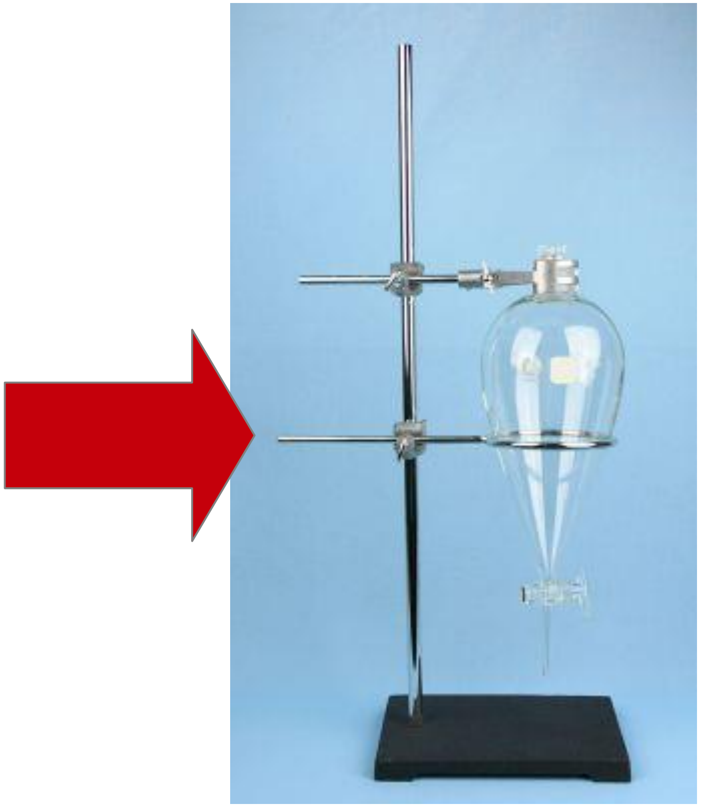
# Reverse Engineering Existing Equipment

Making a simple ring

> Do not design it the way it was made

> For ideal FFF printing you need a solid base on the build platform

> Design for all options for the future

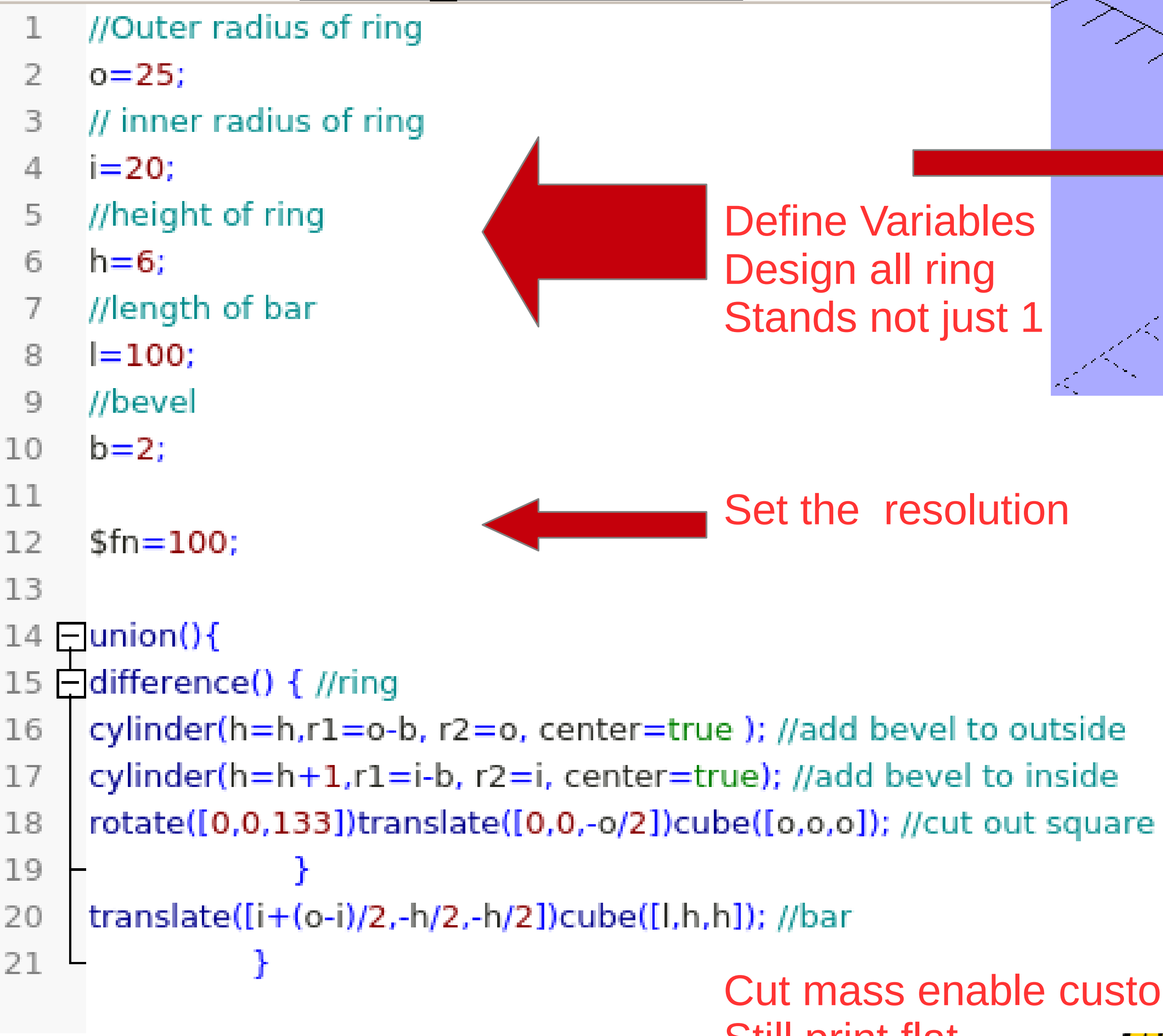


Michigan Tech

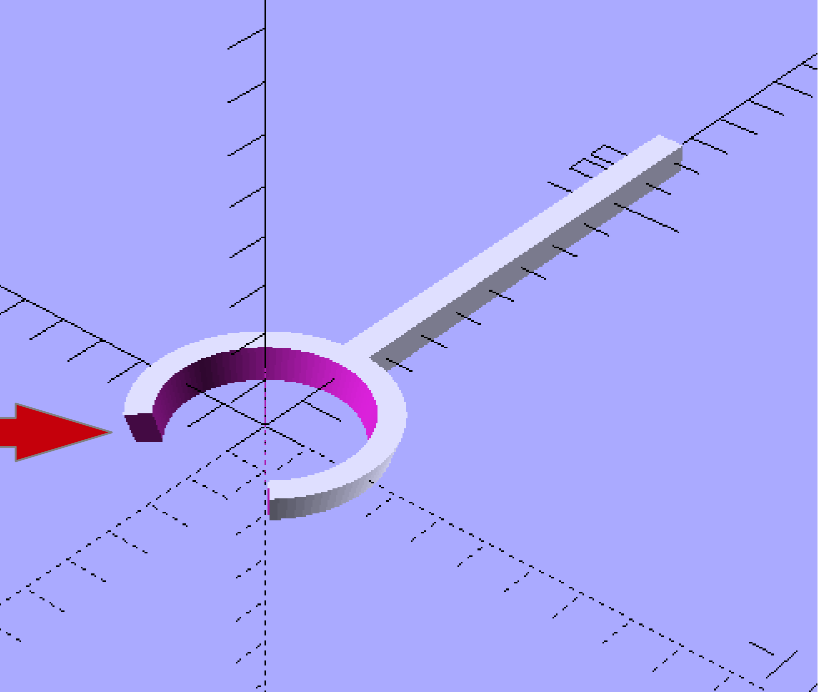
Michigan Technological University

Open Sustainability Technology
Research Group

# Ring Stand - Improved



```
//Outer radius of ring
o=25;
// inner radius of ring
i=20;
//height of ring
h=6;
//length of bar
l=100;
//bevel
b=2;

$fn=100;

union(){
difference() { //ring
cylinder(h=h,r1=o-b, r2=o, center=true ); //add bevel to outside
cylinder(h=h+1,r1=i-b, r2=i, center=true); //add bevel to inside
rotate([0,0,133])translate([0,0,-o/2])cube([o,o,o]); //cut out square
              }
translate([i+(o-i)/2,-h/2,-h/2])cube([l,h,h]); //bar
           }
```



Define Variables
Design all ring
Stands not just 1

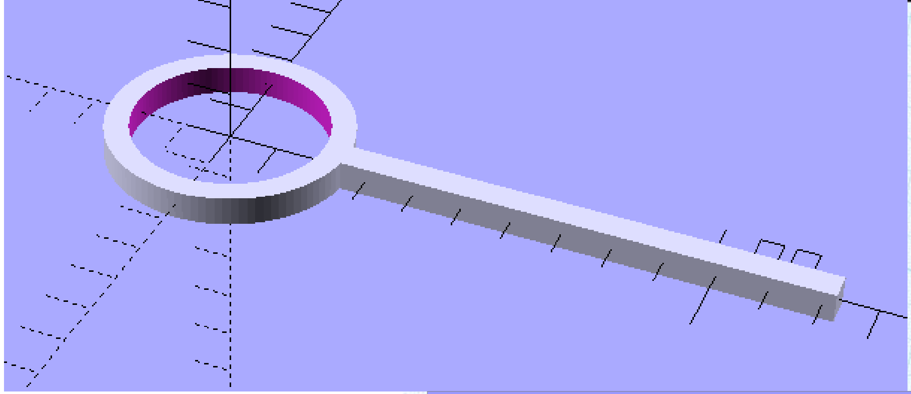
Set the  resolution

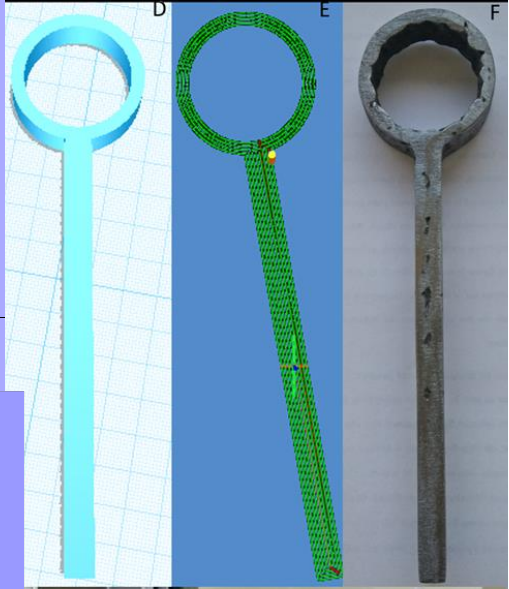
Think about shapes as combina

Cut mass enable custom shapes : b
Still print flat

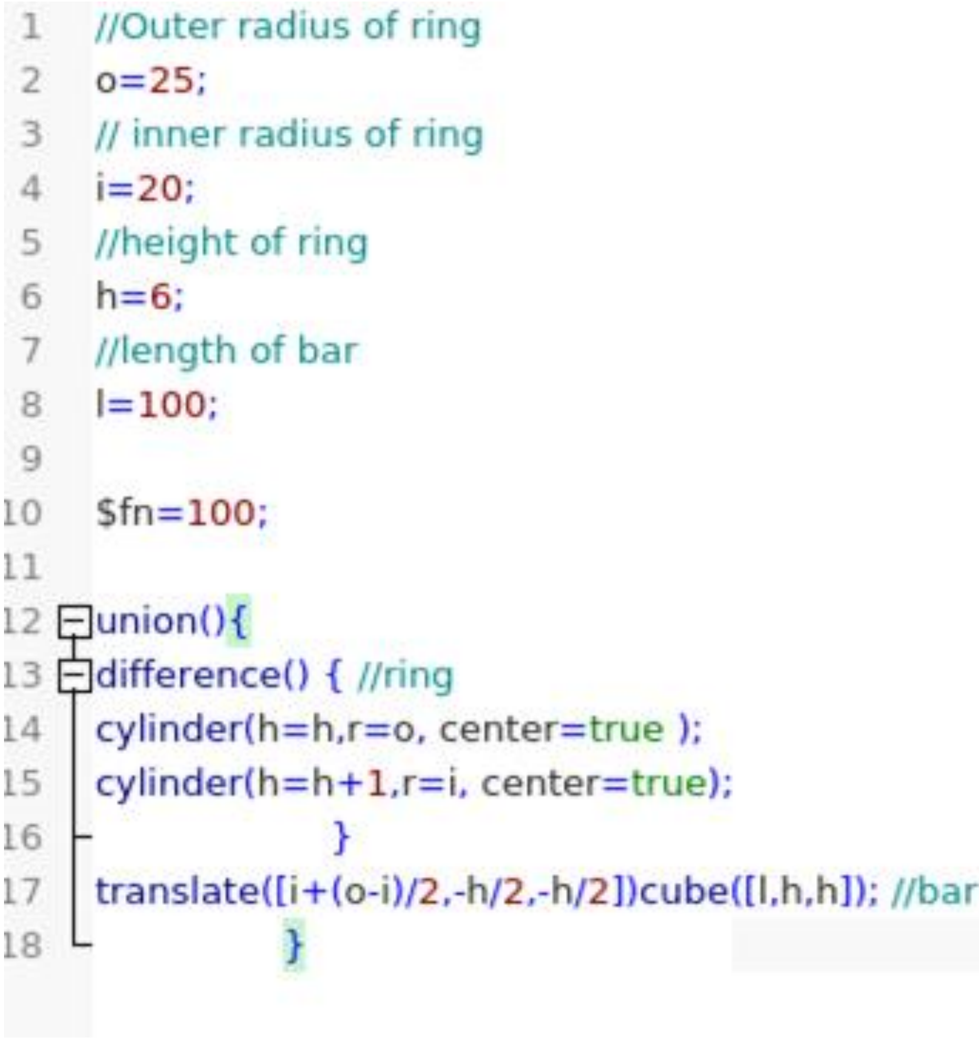
Michigan Tech
Michigan Technological University
Open Sustainability Technology
Research Group

# Ring Stand Applied to Future Printers

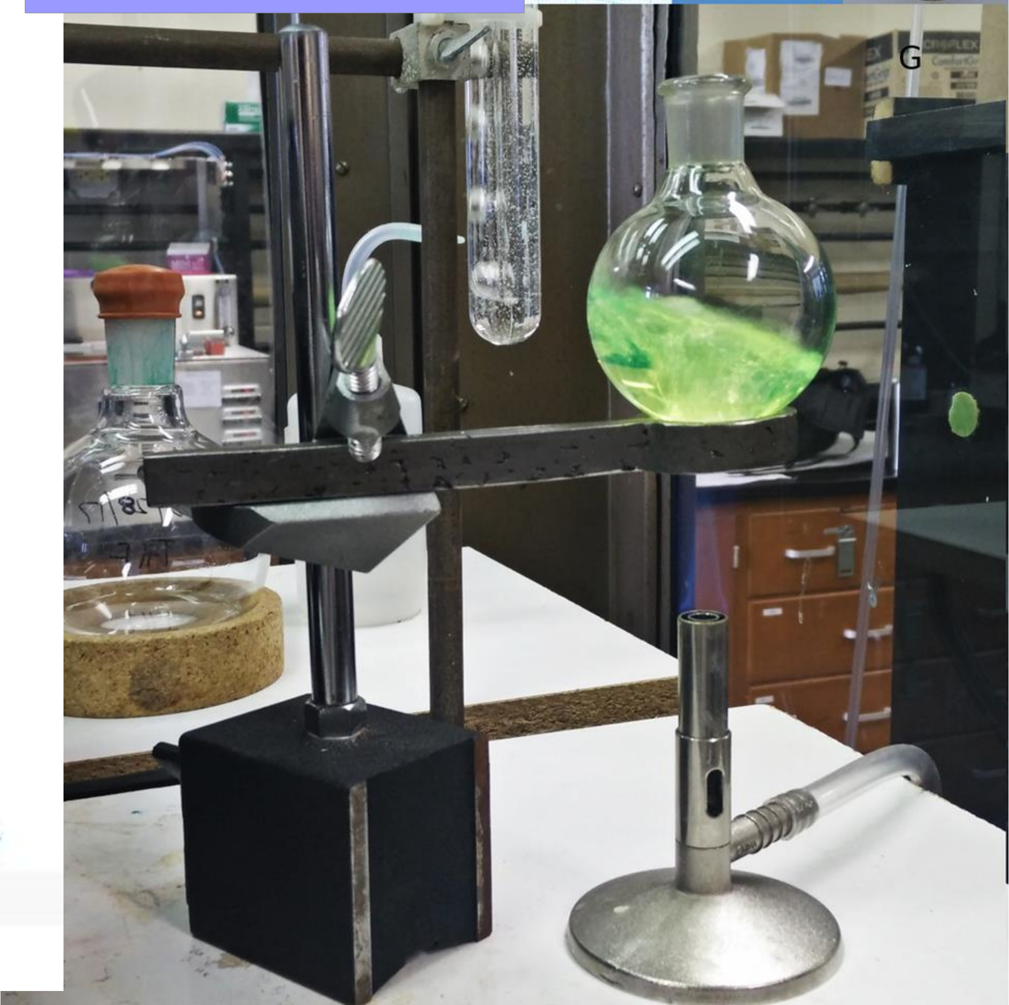




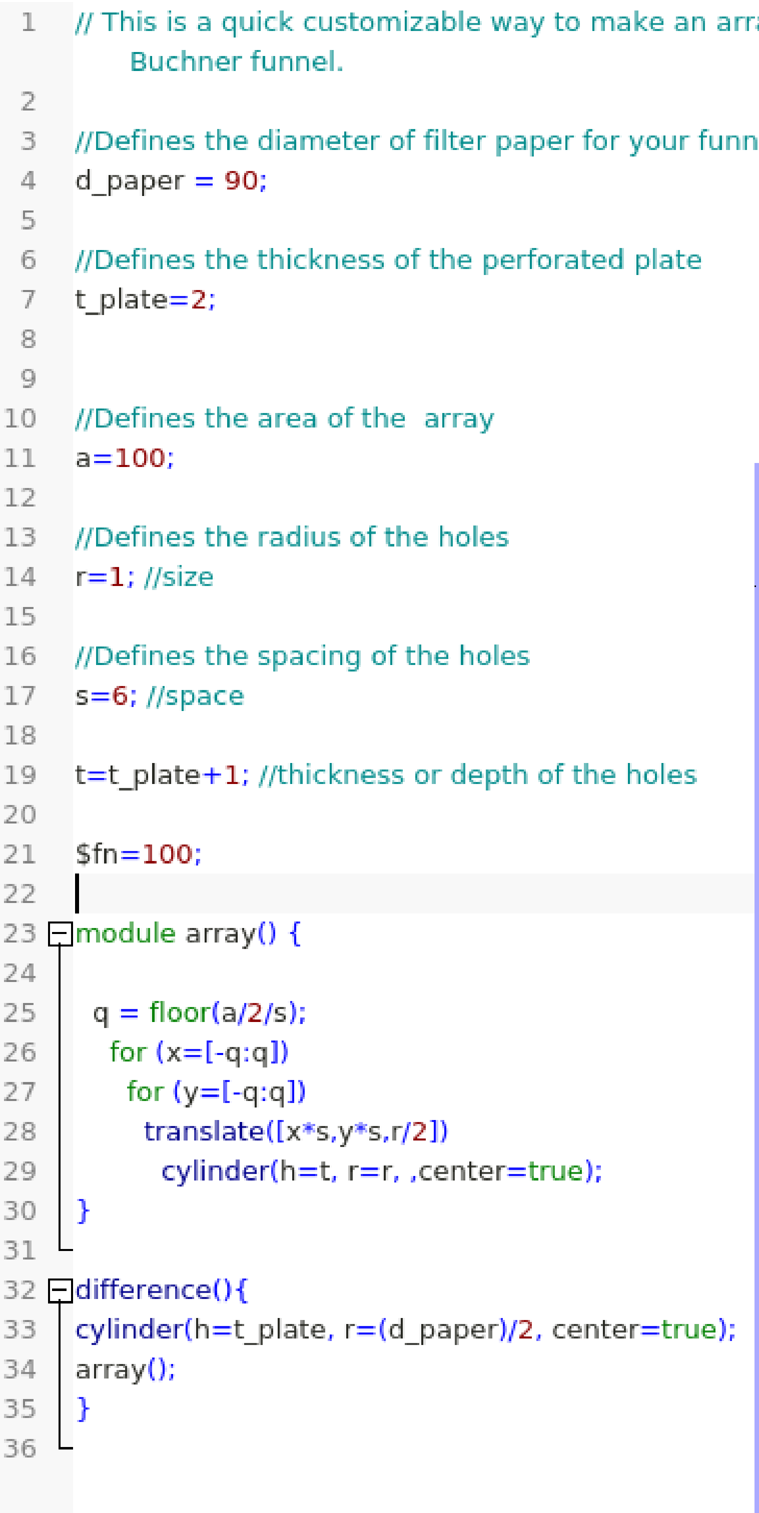
End:
No limits on materials



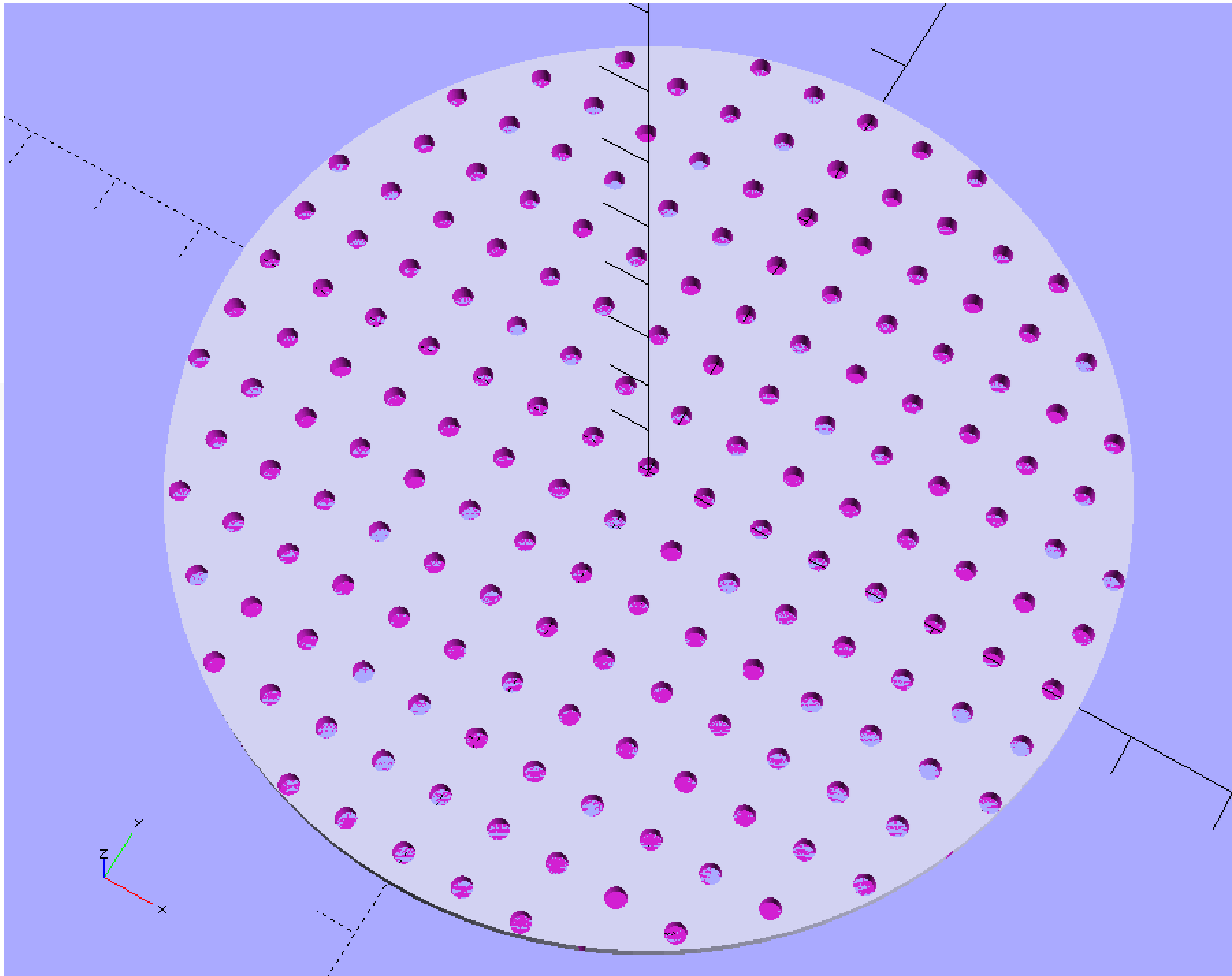
```
//Outer radius of ring
o=25;
// inner radius of ring
i=20;
//height of ring
h=6;
//length of bar
l=100;

$fn=100;

union(){
difference() { //ring
cylinder(h=h,r=o, center=true );
cylinder(h=h+1,r=i, center=true);
                }
translate([i+(o-i)/2,-h/2,-h/2])cube([l,h,h]); //bar
             }
```



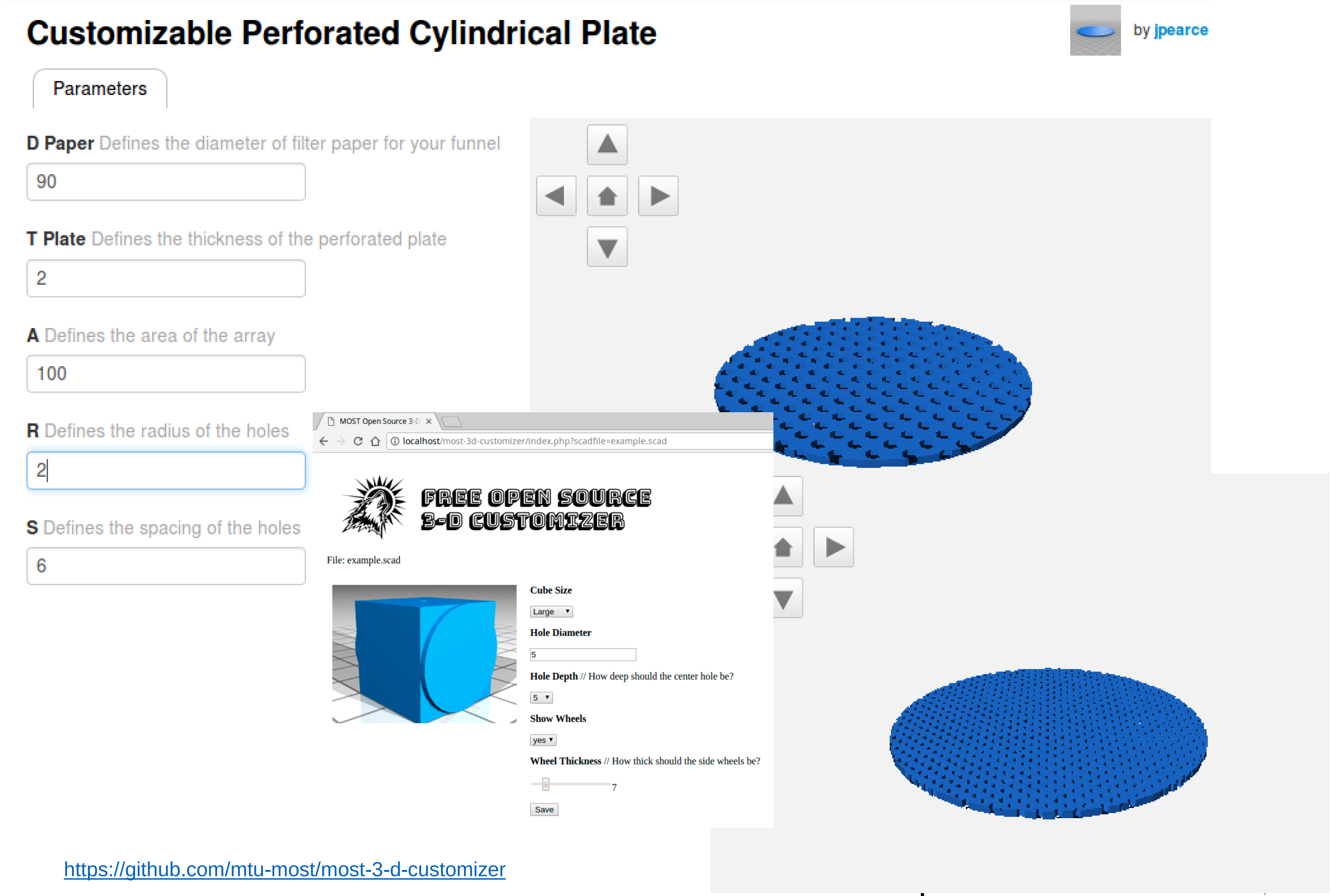
# Plate for Buckner Funnel



```
// This is a quick customizable way to make an array of holes of any size in a cylindrical plate - specifically for use in a Buchner funnel.

//Defines the diameter of filter paper for your funnel
d_paper = 90;

//Defines the thickness of the perforated plate
t_plate=2;


//Defines the area of the  array
a=100;

//Defines the radius of the holes
r=1; //size

//Defines the spacing of the holes
s=6; //space

t=t_plate+1; //thickness or depth of the holes

$fn=100;

module array() {

 q = floor(a/2/s);
  for (x=[-q:q])
   for (y=[-q:q])
    translate([x*s,y*s,r/2])
     cylinder(h=t, r=r, ,center=true);
}

difference(){
cylinder(h=t_plate, r=(d_paper)/2, center=true);
array();
}
```



# Customizer and OS Customizer



## Customizable Perforated Cylindrical Plate

by jpearce

Parameters

**D Paper** Defines the diameter of filter paper for your funnel

90

**T Plate** Defines the thickness of the perforated plate

2

**A** Defines the area of the array

100

**R** Defines the radius of the holes

2

**S** Defines the spacing of the holes

6

MOST Open Source 3-D
localhost/most-3d-customizer/index.php?scadfile=example.scad

FREE OPEN SOURCE 3-D CUSTOMIZER

File: example.scad

**Cube Size**

Large ▾

**Hole Diameter**

5

**Hole Depth** // How deep should the center hole be?

5 ▾

**Show Wheels**

yes ▾

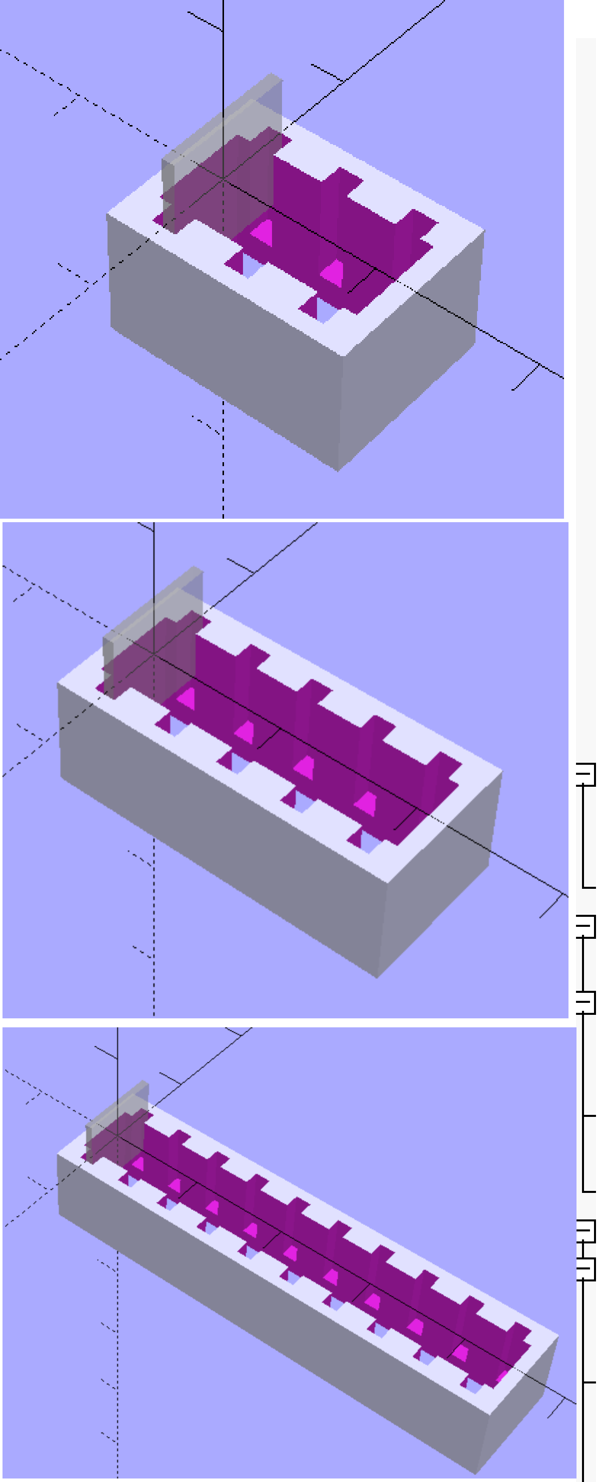
**Wheel Thickness** // How thick should the side wheels be?

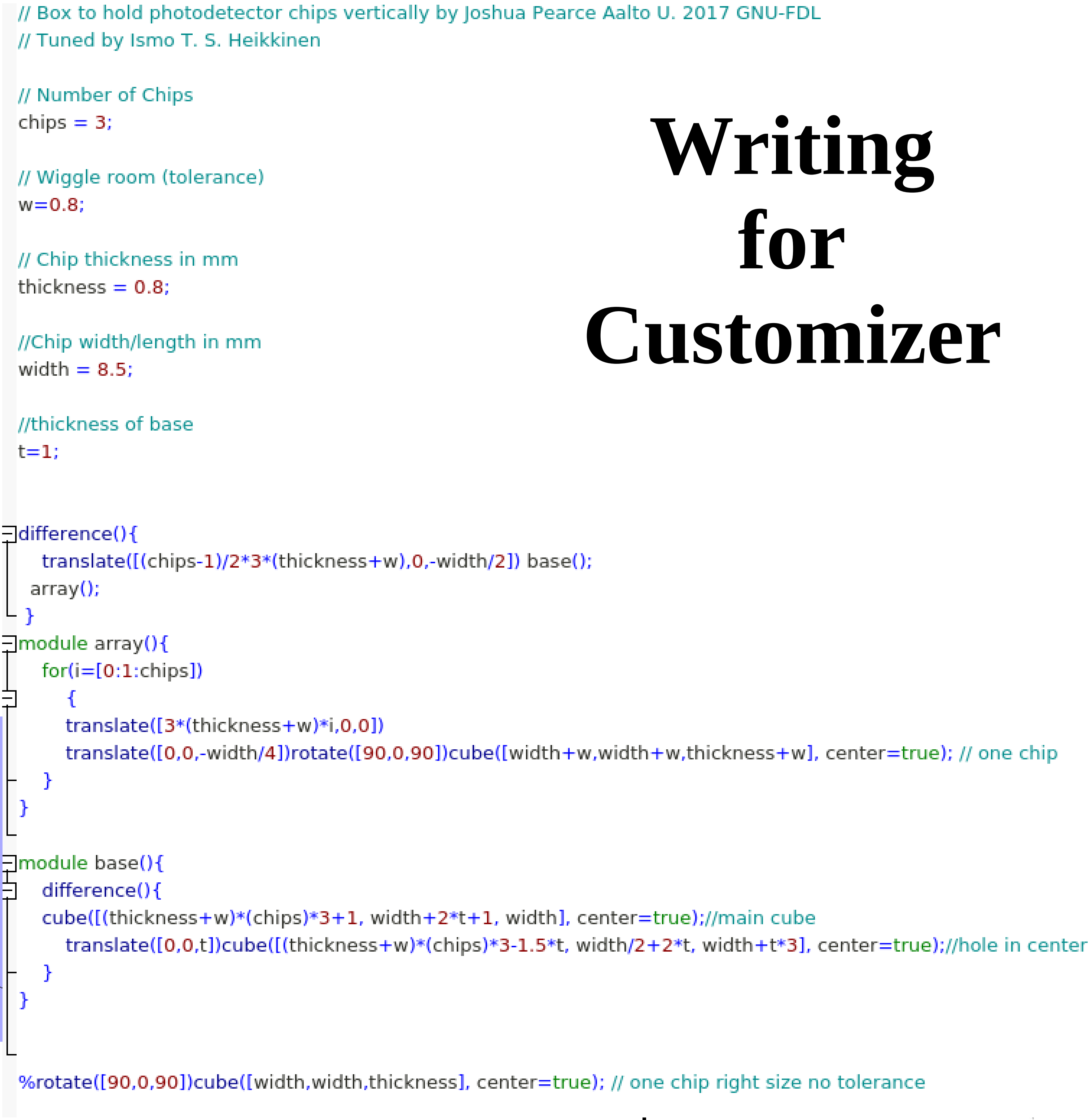
7

Save

https://github.com/mtu-most/most-3-d-customizer

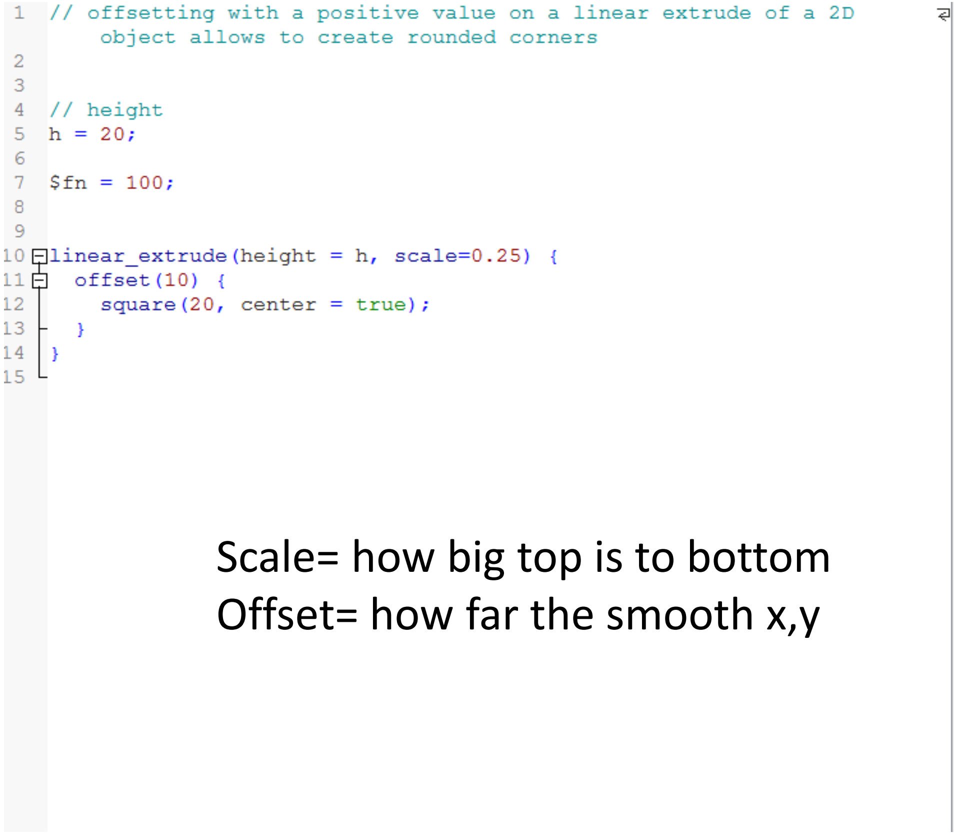
# Writing for Customizer



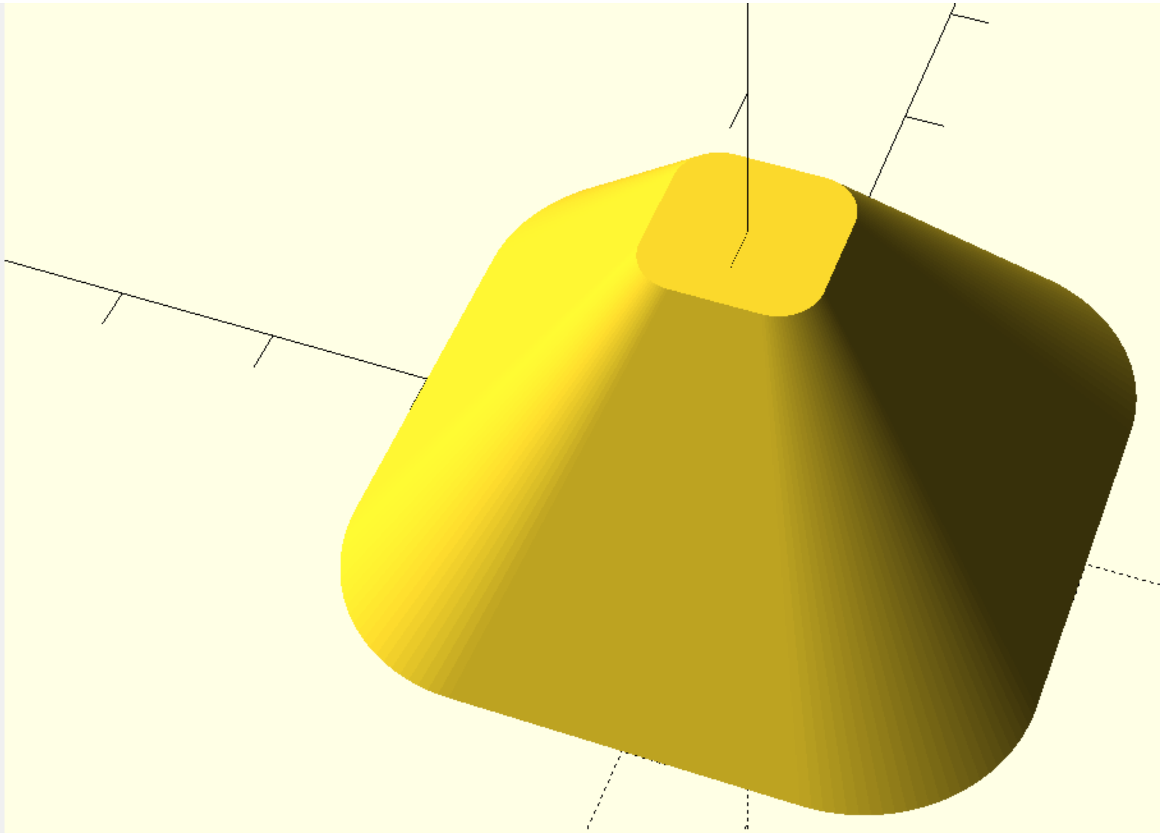


```
// Box to hold photodetector chips vertically by Joshua Pearce Aalto U. 2017 GNU-FDL
// Tuned by Ismo T. S. Heikkinen

// Number of Chips
chips = 3;

// Wiggle room (tolerance)
w=0.8;

// Chip thickness in mm
thickness = 0.8;

//Chip width/length in mm
width = 8.5;

//thickness of base
t=1;



difference(){
    translate([(chips-1)/2*3*(thickness+w),0,-width/2]) base();
  array();
 }
module array(){
    for(i=[0:1:chips])
        {
        translate([3*(thickness+w)*i,0,0])
        translate([0,0,-width/4])rotate([90,0,90])cube([width+w,width+w,thickness+w], center=true); // one chip
    }
}

module base(){
    difference(){
    cube([(thickness+w)*(chips)*3+1, width+2*t+1, width], center=true);//main cube
        translate([0,0,t])cube([(thickness+w)*(chips)*3-1.5*t, width/2+2*t, width+t*3], center=true);//hole in center
    }
}



%rotate([90,0,90])cube([width,width,thickness], center=true); // one chip right size no tolerance
```

# A Few Tricks



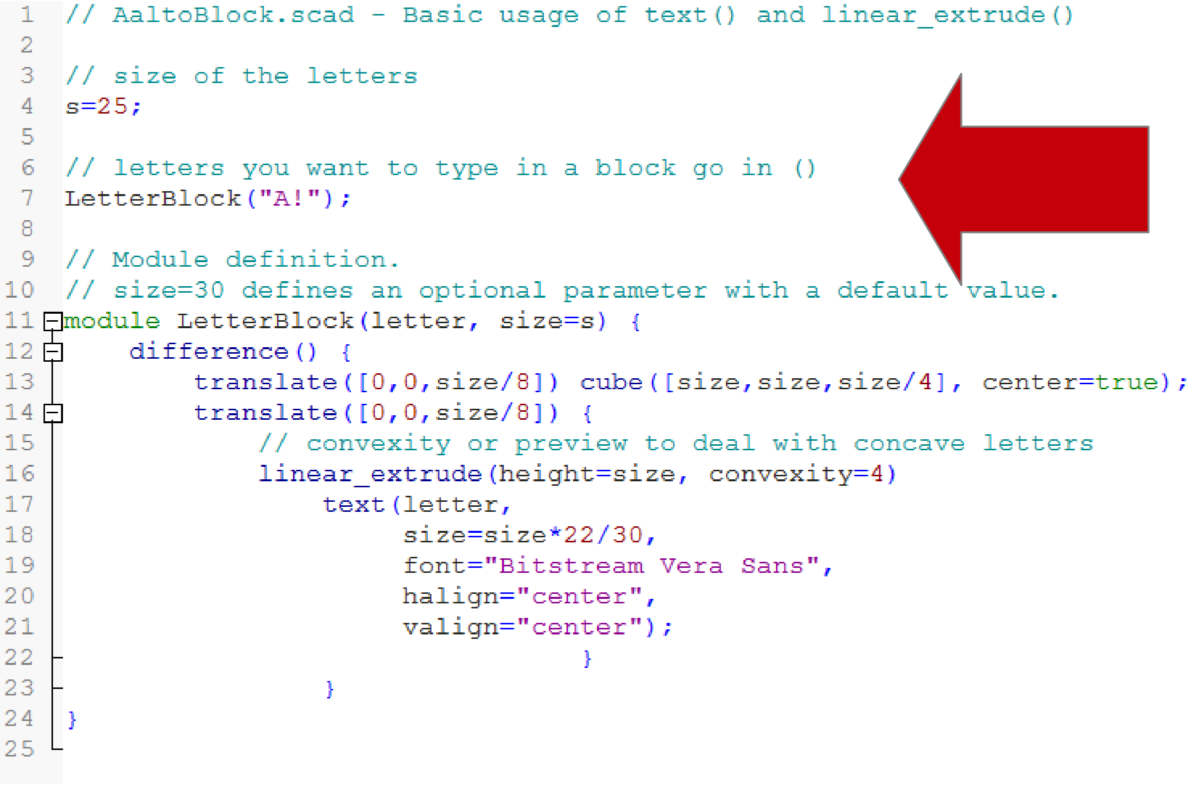
```
// offsetting with a positive value on a linear extrude of a 2D
    object allows to create rounded corners


// height
h = 20;

$fn = 100;


linear_extrude(height = h, scale=0.25) {
  offset(10) {
    square(20, center = true);
  }
}

```



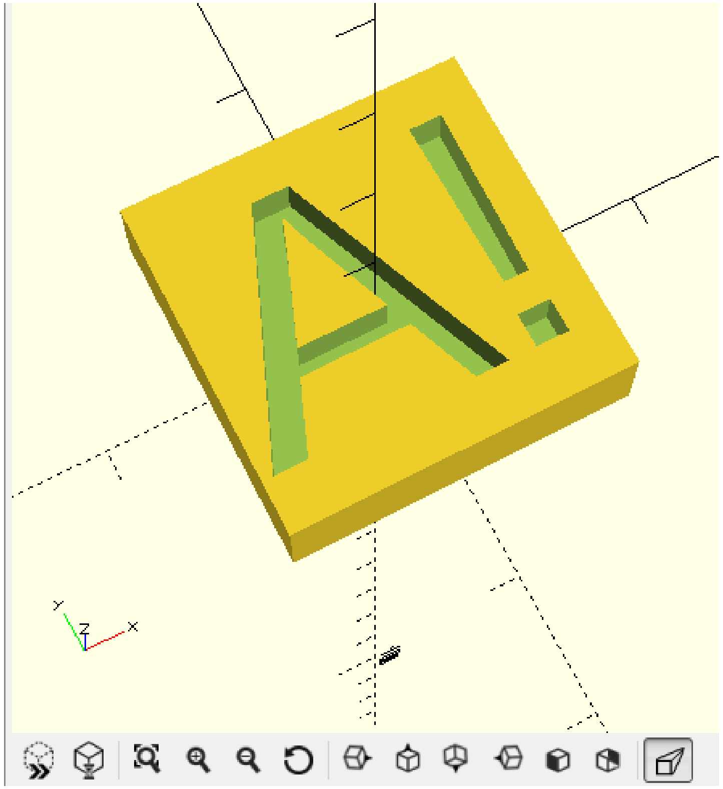
Scale= how big top is to bottom
Offset= how far the smooth x,y

Michigan Tech
Michigan Technological University
Open Sustainability Technology
Research Group

# Customize:Aalto Block



```
// AaltoBlock.scad - Basic usage of text() and linear_extrude()

// size of the letters
s=25;

// letters you want to type in a block go in ()
LetterBlock("A!");

// Module definition.
// size=30 defines an optional parameter with a default value.
module LetterBlock(letter, size=s) {
    difference() {
        translate([0,0,size/8]) cube([size,size,size/4], center=true);
        translate([0,0,size/8]) {
            // convexity or preview to deal with concave letters
            linear_extrude(height=size, convexity=4)
                text(letter,
                     size=size*22/30,
                     font="Bitstream Vera Sans",
                     halign="center",
                     valign="center");
                                }
                }
}
```

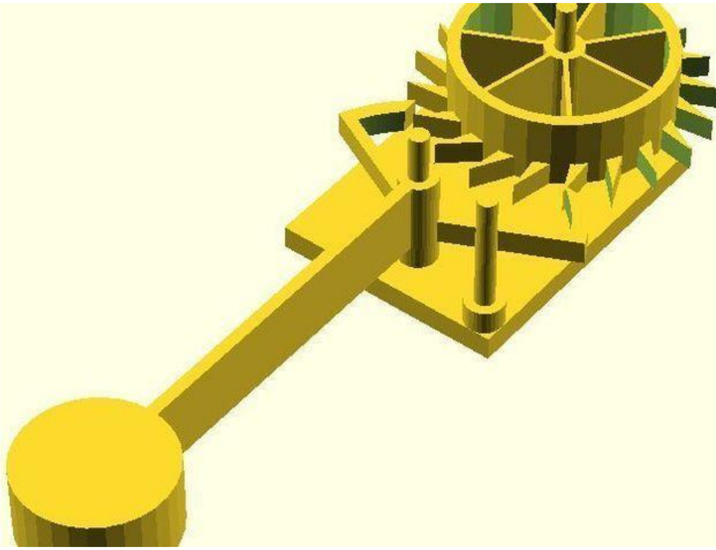


Michigan Tech
Michigan Technological University
Open Sustainability Technology
Research Group

# Use Past Work

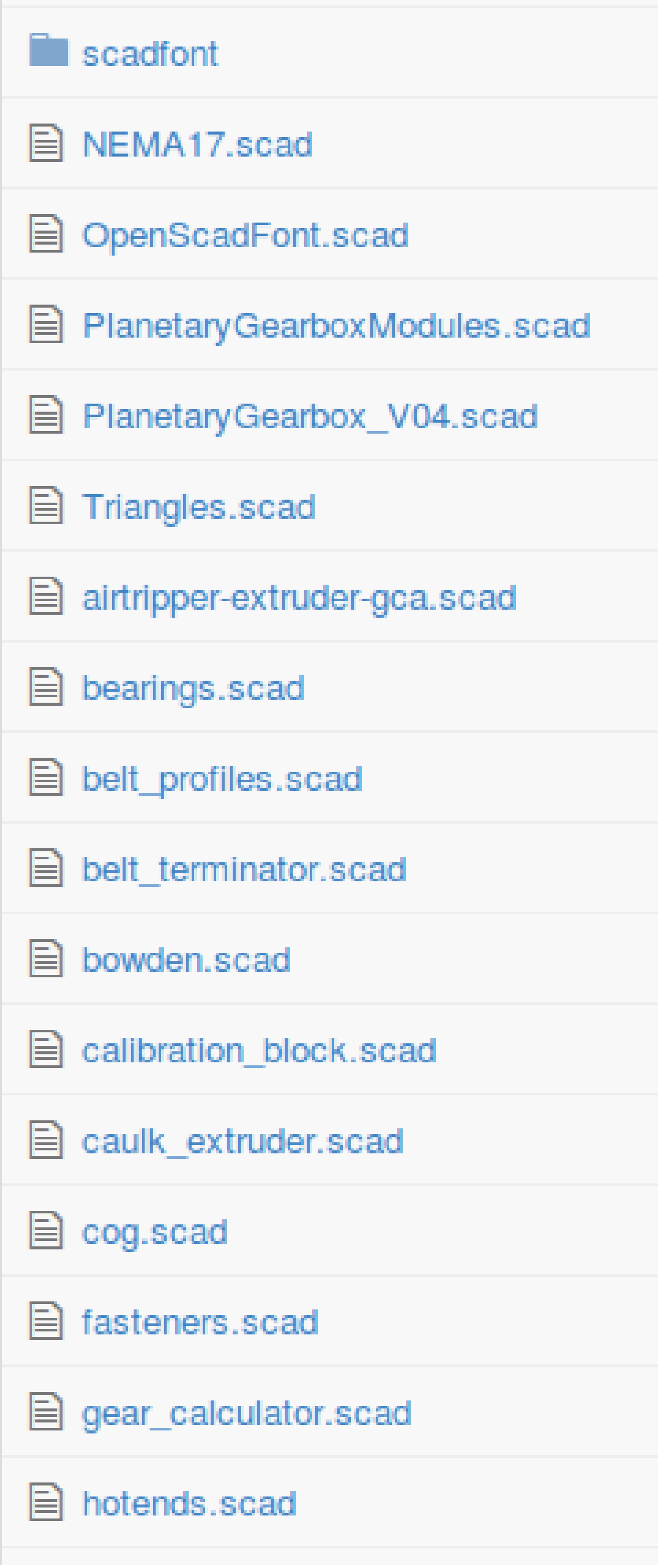
**Libraries:**

```
use <MCAD/involute_gears.scad>
include <escapementLibrary.scad>
```



You are using collections of
Modules written before...
Or pre-defined variables

# MOST Lab Libraries on Github



scadfont

NEMA17.scad

OpenScadFont.scad

PlanetaryGearboxModules.scad

PlanetaryGearbox_V04.scad

Triangles.scad

airtripper-extruder-gca.scad

bearings.scad

belt_profiles.scad

belt_terminator.scad

bowden.scad

calibration_block.scad

caulk_extruder.scad

cog.scad

fasteners.scad

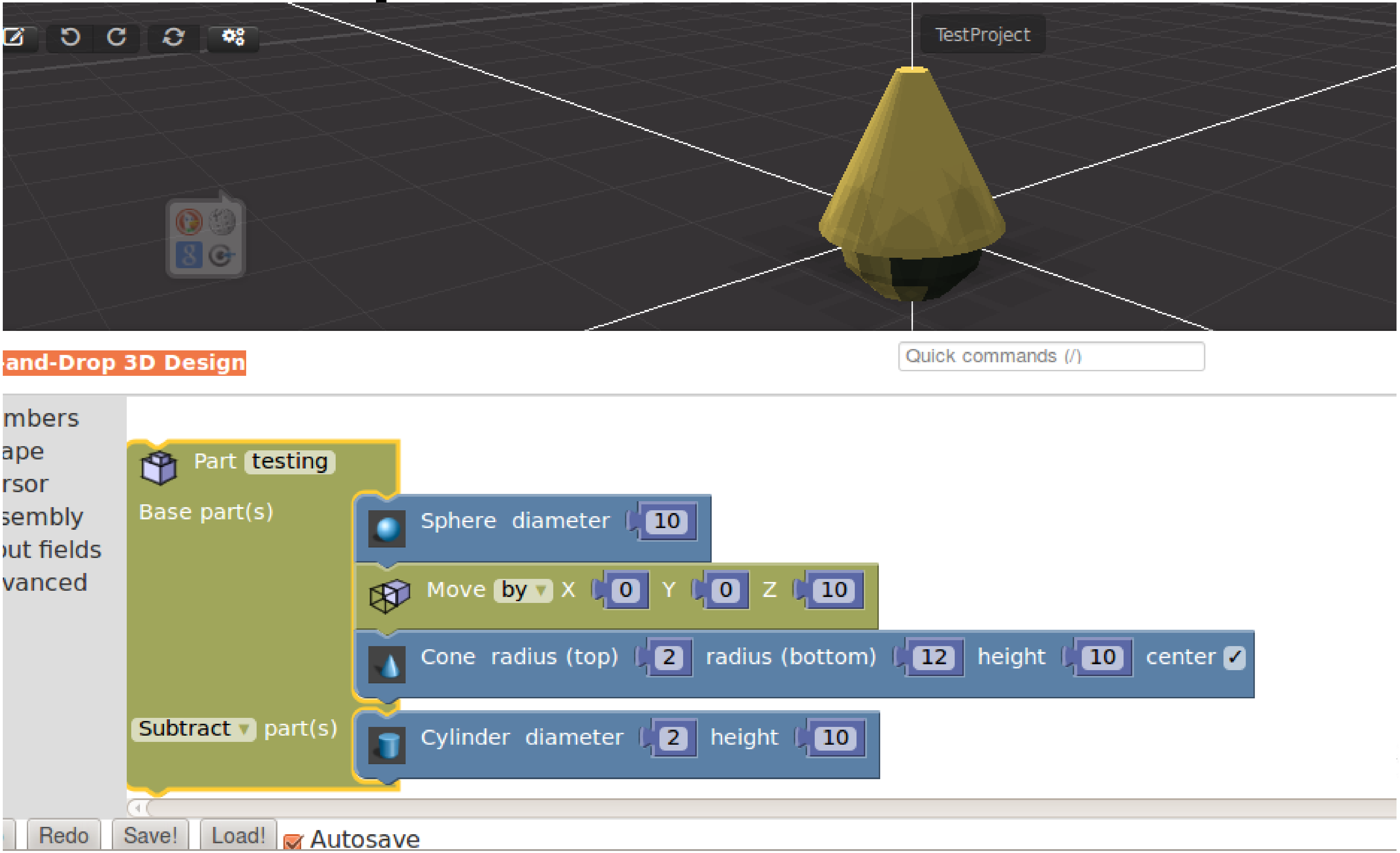
gear_calculator.scad

hotends.scad

- Do not re-invent the wheel
- Stand on the Shoulders of Giants
- Collection of the most useful libraries written at MTU and elsewhere
- https://github.com/mtu-most/most-scad-libraries

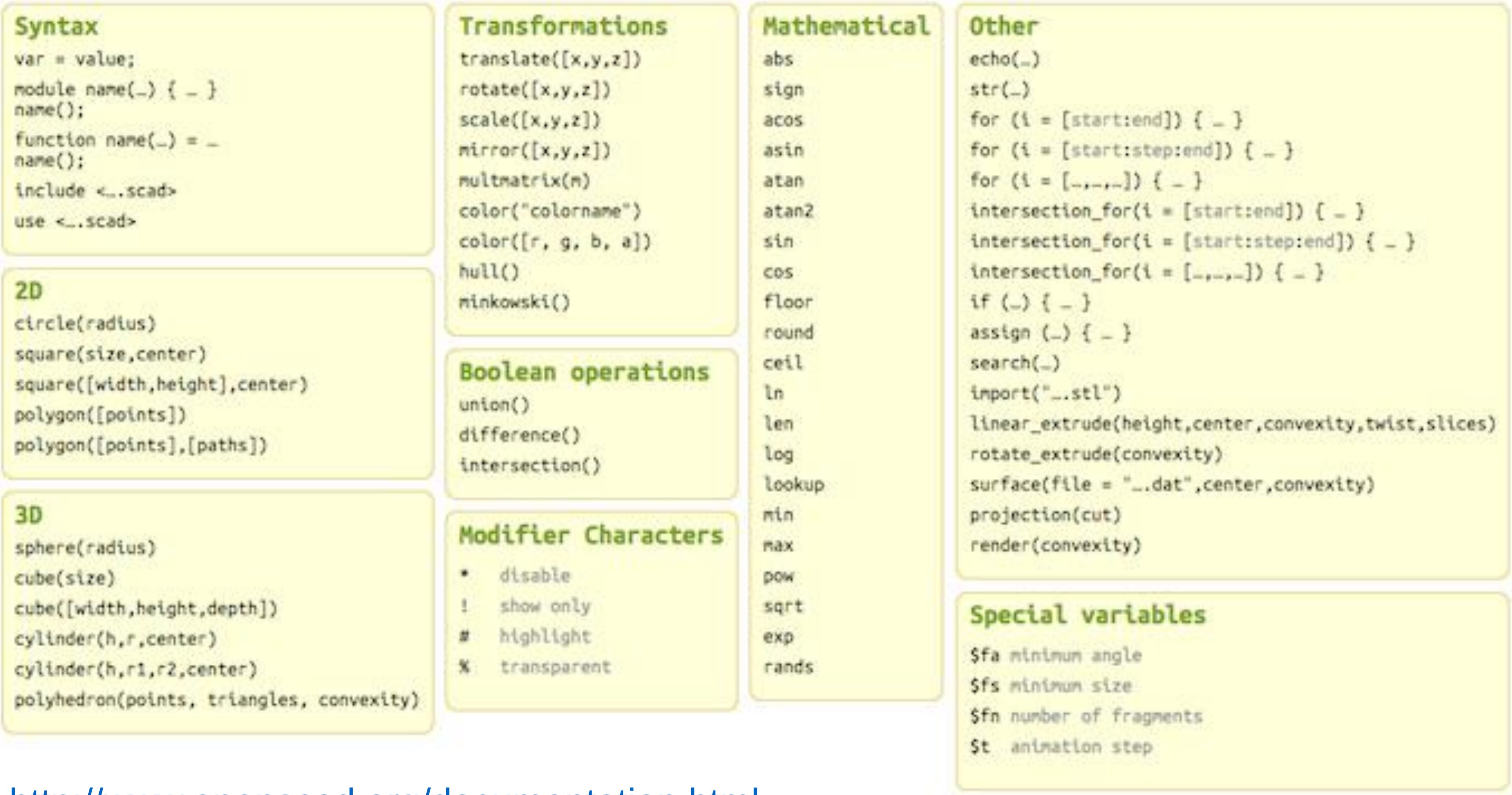
Michigan Tech
Michigan Technological University
Open Sustainability Technology
Research Group

# What if I can't type?
# Object Oriented SCAD
# SnapSCAD or UltiCreator



```
Part testing
Base part(s)
    Sphere diameter 10
    Move by X 0 Y 0 Z 10
    Cone radius (top) 2 radius (bottom) 12 height 10 center ✓
Subtract part(s)
    Cylinder diameter 2 height 10
```

# Cheat Sheet



### Syntax

```
var = value;
module name(…) { … }
name();
function name(…) = …
name();
include <….scad>
use <….scad>
```

### 2D

```
circle(radius)
square(size,center)
square([width,height],center)
polygon([points])
polygon([points],[paths])
```

### 3D

```
sphere(radius)
cube(size)
cube([width,height,depth])
cylinder(h,r,center)
cylinder(h,r1,r2,center)
polyhedron(points, triangles, convexity)
```

### Transformations

```
translate([x,y,z])
rotate([x,y,z])
scale([x,y,z])
mirror([x,y,z])
multmatrix(m)
color("colorname")
color([r, g, b, a])
hull()
minkowski()
```

### Boolean operations

```
union()
difference()
intersection()
```

### Modifier Characters

```
*   disable
!   show only
#   highlight
%   transparent
```

### Mathematical

```
abs
sign
acos
asin
atan
atan2
sin
cos
floor
round
ceil
ln
len
log
lookup
min
max
pow
sqrt
exp
rands
```

### Other

```
echo(…)
str(…)
for (i = [start:end]) { … }
for (i = [start:step:end]) { … }
for (i = […,…,…]) { … }
intersection_for(i = [start:end]) { … }
intersection_for(i = [start:step:end]) { … }
intersection_for(i = […,…,…]) { … }
if (…) { … }
assign (…) { … }
search(…)
import("….stl")
linear_extrude(height,center,convexity,twist,slices)
rotate_extrude(convexity)
surface(file = "….dat",center,convexity)
projection(cut)
render(convexity)
```

### Special variables

```
$fa minimum angle
$fs minimum size
$fn number of fragments
$t  animation step
```

http://www.openscad.org/documentation.html

Michigan Tech
Michigan Technological University
Open Sustainability Technology
Research Group

## More information

- http://www.openscad.org/
- http://en.wikibooks.org/wiki/OpenSCAD_User_Manual
- http://www.appropedia.org/MOST
- http://reprap.org/

Michigan Tech
Michigan Technological University
Open Sustainability Technology
Research Group